



Assignment of bachelor's thesis

Title:	Automated data drift analysis using great expectations library
Student:	Michal Melko
Supervisor:	Ing. Daniel Vařata, Ph.D.
Study program:	Informatics
Branch / specialization:	Knowledge Engineering
Department:	Department of Applied Mathematics
Validity:	until the end of summer semester 2023/2024

Instructions

Every machine learning project deployed in a real-world environment always faces problems with constantly updating datasets. This brings a number of pitfalls, mainly a risk that data on which model ought to be used (for making predictions or updating the model itself) does not meet the assumptions the model has been built with.

There can be several problems - from missing columns through changes in data (distribution of variables, missing levels, the ratio of missing values, etc.) to different correlations between variables. Usage of this model can therefore lead to erroneous and misleading results. Because of this, it is highly recommended to deal with data quality in models deployed to production and, if possible, reveal potential problems in time.

One of the tools that can address the abovementioned issues is Python library Great Expectations. It is a framework that enables the creation of a set of expectations and checks their fulfillment for new datasets. Nonetheless, one arrives at certain limitations of this library, like the absence of multivariable data drift detection methods and the presence of only a very scarce tool for automatic expectations builder (so-called profiling) based on a given dataset.

As a part of the thesis, the student will be extending the framework Great Expectations in both aforementioned areas.

Based on the research of known methods and approaches, the student will suggest and implement a set of expectations, allowing the user to detect data drift in multiple



**FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE**

dimensions (e.g., using PCA projection, Chi-square test, bootstrap, etc., depending on the variable type). With their help and using additional expectations provided by the framework, he will then create a profiler, i.e., an automated tool that, based on the given dataset, creates a suitable set of expectations for basic data testing. He will take into account the distribution of individual variables, the size of a dataset, etc. The functionality and contribution of the thesis will be demonstrated on at least two suitable datasets, where the benefits of the implemented methods will be clearly shown.



Bachelor's thesis

**AUTOMATED DATA
DRIFT ANALYSIS USING
GREAT EXPECTATIONS
LIBRARY**

Michal Melko

Faculty of Information Technology
Katedra teoretické informatiky
Supervisor: Ing. Daniel Vařata, Ph.D.
February 15, 2024

Czech Technical University in Prague
Faculty of Information Technology

© 2024 Michal Melko. All rights reserved.

This thesis is school work as defined by Copyright Act of the Czech Republic. It has been submitted at Czech Technical University in Prague, Faculty of Information Technology. The thesis is protected by the Copyright Act and its usage without author's permission is prohibited (with exceptions defined by the Copyright Act).

Citation of this thesis: Melko Michal. *Automated data drift analysis using great expectations library*. Bachelor's thesis. Czech Technical University in Prague, Faculty of Information Technology, 2024.

Contents

Acknowledgments	v
Declaration	vi
Abstract	vii
List of abbreviations	viii
Introduction	1
1 Theory	3
1.1 Definitions	3
1.1.1 Data drift	3
1.1.2 Multivariate data drift	6
2 Research	9
2.1 Great Expectations	9
2.2 NannyML	9
2.3 Evidently	10
3 Methods discussion	11
3.1 Univariate drift detection	11
3.1.1 Kolmogorov–Smirnov test	11
3.1.2 Kullback-Leibler divergence	11
3.1.3 Population Stability Index	12
3.2 Multivariate drift detection	12
3.2.1 Binary classifier	12
3.2.2 Clustering	12
3.2.3 Gaussian Mixture Models(GMM)	13
3.2.4 Principal Component Analysis (PCA)	13
3.3 Methods comparison	13
3.3.1 No drift	15
3.3.2 Univariate shift - mean	16
3.3.3 Univariate shift - standard deviation	17
3.3.4 Univariate shift - mean and standard deviation	18
3.3.5 Multivariate shift - switching signs	19
3.3.6 Multivariate shift - change in scale	21
3.3.7 Adding outliers	22
4 Results	24
5 Implementation	25
5.1 Expectation	25
5.2 Profiler	25

Contents

iii

6 Conclusion 26

Contents of the attachment 29

List of Figures

1.1	Anscombe's quartet [4]	8
3.1	Tested and referential dataset comparison without drift	15
3.2	Tested and referential dataset comparison with shift in mean	16
3.3	Tested and referential dataset comparison with shift in standard deviation	18
3.4	Tested and referential dataset comparison with shift in mean and standard deviation	19
3.5	Tested and referential dataset comparison with multivariate shift - just the sign	20
3.6	Tested and referential dataset comparison with multivariate shift - just the scale	21
3.7	Tested and referential dataset comparison with adding outliers	22

List of Tables

1.1	Example students dataset	5
1.2	Dataset after data drift	5
1.3	Dataset after concept drift	6
1.4	Dataset after multivariate drift	7
3.1	No drift	15
3.2	Univariate drift - shift in mean	17
3.3	Univariate drift - shift in standard deviation	17
3.4	Univariate drift - shift in mean and standard deviation	19
3.5	Multivariate shift - just the sign	20
3.6	Adding outliers	23

List of code listings

Foremost, I would like to thank Mgr. Dominik Matula for his immensely valuable advice, astounding patience and unwavering resolve in seeing this thesis finished. Next I would like to thank Ing. Daniel Vařata, Ph.D. for supervising this thesis on behalf of Faculty of Information Technology, CTU in Prague.

Declaration

I hereby declare that the presented thesis is my own work and that I have cited all sources of information in accordance with the Guideline for adhering to ethical principles when elaborating an academic final thesis.

I acknowledge that my thesis is subject to the rights and obligations stipulated by the Act No. 121/2000 Coll., the Copyright Act, as amended, in particular the fact that the Czech Technical University in Prague has the right to conclude a licence agreement on the utilization of this thesis as a school work pursuant of Section 60 (1) of the Act.

In Praze on February 15, 2024

Abstract

This thesis concerns itself with the data drift detection. There are already existing and widely used data quality tools, one of them being Great expectations library. Among its features at the time of writing this thesis, multivariate drift detection is not present. The thesis compares different approaches that can be used for this purpose. Based on the results of this comparison, Principal component analysis is used to extend the features of the Great expectation library.

Keywords datový drift, knihovna jazyka Python, knihovna Great Expectations, vícerozměrný posun v datech, analýza hlavních komponent, detekce posunu v datech, rozšíření knihovny, kvalita dat

Abstrakt

Tato práce se zabývá detekcí změn v datech. Pro zjišťování kvality dat již existují používané nástroje, jedním z nich je knihovna Great expectations. Mezi jejími funkcemi v době psaní této práce není přítomna detekce vícerozměrného driftu. Práce porovnává různé přístupy, které lze k tomuto účelu použít. Na základě výsledků tohoto srovnání je k rozšíření funkcí knihovny Great expectation použita analýza hlavních komponent.

Klíčová slova data drift, Python library, Great Expectations library, multivariate data drift, principal component analysis, drift detection, library extension, data quality

List of abbreviations

GX	Great Expectations (Python library)
PCA	Principal component analysis
KS	Kolmogorov–Smirnov test
DDD	Data drift detection
MDD	Multivariable data drift

Introduction

Every machine learning project deployed in a real-world environment always faces problems with constantly updating datasets. This brings a number of pitfalls, mainly a risk that data on which model ought to be used (for making predictions or updating the model itself) does not meet the assumptions the model has been built with. There can be several problems - from missing columns through changes in data (distribution of variables, missing levels, the ratio of missing values, etc.) to different correlations between variables. Usage of this model can therefore lead to erroneous and misleading results. Because of this, it is highly recommended to deal with data quality in models deployed to production and, if possible, reveal potential problems in time. One of the most used tools at the time of writing this thesis, that can address the abovementioned issues is a Python library Great Expectations(GX) [1]. It is a framework that enables the creation of a set of expectations and checks their fulfillment for new datasets. Nonetheless, one arrives at certain limitations of this library, like the absence of multivariate data drift detection methods and the presence of only a very scarce tool for automatic expectations builder (so-called profiling) based on a given dataset.

Objectives

First goal of the thesis is to analyse methods for multivariate data drift detection and compare their strengths and weaknesses. Finally based on the result choose the appropriate ones for new so-called expectation in GX library.


Second goal is to create datasets on which this multivariate data drift detection method can be tested and validated. Datasets should be diverse and cover enough of edge cases to show, that chosen method (or methods) are suitable.

Third goal is to use this new expectation together with the existing ones and create a profiler, which will be able to construct a set of expectations according to the given dataset.

Thesis overview

1. Mathematical background
2. Great expectations
3. Methods discussion
4. Implementation
5. Results

6. Conclusion



Chapter 1

Theory

Perhaps the most known parts of the machine learning process are data exploration and model training. Although they are without doubt very important, the process is not done once the model is successfully created and put into production. In many cases, new data is presented to a functioning model in batches. The problem occurs when the model fails to achieve the expected accuracy on some batch. Without further investigation, it is difficult to say whether there is some hidden flaw in the model or whether the problem lies in the new data. The model was built with some assumptions and if they are not met, it is misleading to interpret its results. However, to determine when they are not met is difficult for the model's user and it may pass unnoticed. This is where the data quality checks come in. By running them periodically on the new data, possible discrepancies and deviations from initial assumptions are discovered early on and do not lead to model misuse and unnecessary dismantling of model in search for flaws, that are nonexistent. Data drift detection is an important part of the data quality checks and is also the main topic of this thesis.

1.1 Definitions

There are multiple naming conventions, when it comes to machine learning. Therefore it is appropriate to establish some definitions.

We will call **features**, **feature variables** or **predictors** those variables, that serve as an input to the model. Variable, which is predicted based on them will be called **target**, or **response** variable. The model's output is then the estimate of the target.

In this theses we will mainly concern ourselves with continuous variables.

1.1.1 Data drift

Data drift is a well known term in machine learning community. However, during the research for this thesis we found out, that different papers referred to it with different names (e.g. data drift, virtual drift, covariate shift). Therefore, we will put forward some definitions to avoid confusion.

In terms of probability, we can define **data drift** as the change in the distribution of the predictor variables, meaning $P(X) \neq P(X')$, where X denotes training data and X' denotes production data.

This happens independently of the target variable. In many real world applications, the target variable depends not only on known predictor variables but also on some hidden concept that is unknown to us at the time of model training. Change to this hidden concept is what

we call a **concept drift**. In this case relationship between predictor variables and the target changes, i.e. $P(y|X)$ changes. An example would be model predicting weather for a given day. If we took into account only weather variables from previous few weeks, like temperature, air pressure, humidity, etc., our model predictions would fall apart once the seasons change. In this case, season was our hidden concept and change to it thus created a concept drift. These two types of drift often occur at the same time.

There are multiple categories commonly distinguished in literature[2]. We can divide the data drift into categories by:

1. Period over which it occurs

Here we can divide data drift into sudden, gradual and recurring. **Sudden data drift** occurs over short period of time. Staying with our example of weather prediction, a volcanic eruption the size of 1883 eruption of Krakatoa can cause a volcanic winter, decreasing the global temperature. This in turn would throw off our model's predictions, because model did not take this event into account during training.

Gradual data drift occurs over longer periods of time. An example would be the change in temperature caused by global warming, whether caused by human activities, or variations in Earth's orbit.

Recurring drift occurs periodically. This can be represented for example by the changing of the seasons. In this case, if we are aware of this concept, we can save the description of the drift for each period and use it to correct our model.

2. The manner of occurrence

Drift does not have to occur only because data, on which we are basing our predictions are out of date. It can also happen because the data are from different *location*. Therefore it can be useful to distinguish between **drift over time** and **drift over environment**. Drift over time is what we were talking about up to this point. This encompasses a hidden concept, changing with time. Drift over environment happens, when we try to use our model on data from different environment, than what we used in training. In line with our example, this could mean trying to use the weather prediction model trained on data from rainforests to predict weather on poles. Or on Mars, for that matter.

1.1.1.1 Data drift example

In order to better visualise and get hold of the idea of the data drift and the concept drift, let us create a concrete example. Faculty, under which this thesis is written resides in building shared with another faculty of the same university – Faculty of architecture. Assume a model, which takes a look at a student entering the said building and makes a prediction, whether he belongs to the Faculty of Information Technology (FIT), or Faculty of Architecture (FA). This is an example of a classification problem. Let us say that the model has two features at its disposal, students gender and whether he is carrying drawing boards. An example training dataset could look like the one in the table 1.1.

faculty	gender	has boards
FIT	F	1
FIT	M	0
FIT	M	0
FIT	M	0
FA	F	0
FA	F	0
FA	F	1
FA	F	1
FA	M	1
FA	M	1

■ **Table 1.1** Example students dataset

faculty	gender	has boards
FIT	F	1
FIT	M	0
FIT	M	0
FIT	M	0
FA	F	0
FA	M	1
FA	M	1

■ **Table 1.2** Dataset after data drift

Now, we can make a presumption, that drawing boards will be more often carried by students of architecture, than students of computer science. Let us compute the probabilities from the given data.

We will define B as an event *student has drawing boards*. Event FA will be defined as *student is studying at Faculty of Architecture*. Then,

$$P(B) = \frac{1}{2} = 0.5$$

$$P(FA) = \frac{3}{5} = 0.6$$

$$P(B|FA) = \frac{2}{3} = 0.\bar{6}$$

$$P(FA|B) = \frac{4}{5} = 0.8$$

An example of data drift would be, if most of FA lectures moved to another building, leading to decrease in number of FA students.

Production dataset could then look like in the table 1.2.

The probabilities in question would then be

$$P(B) = \frac{3}{7} \approx 0.429$$

$$P(FA) = \frac{3}{7} \approx 0.429$$

$$P(B|FA) = \frac{2}{3} = 0.\bar{6}$$

$$P(FA|B) = \frac{2}{3} = 0.\bar{6} \tag{1.1}$$

As we can see, probability that student is from FA based on whether he is carrying the drawing boards or not has decreased (equation 1.1). If we had model built on this presumption, its accuracy would significantly decrease because of the data drift. Note that relationship between carrying drawing desks and being from FA, i.e. $P(B|FA)$, did not change. Assumption still holds, just the data drift caused less imbalance between predictor variables.

faculty	gender	has boards
FIT	F	1
FIT	M	1
FIT	M	1
FIT	M	1
FA	F	0
FA	F	0
FA	F	0
FA	F	0
FA	M	0
FA	M	1

■ **Table 1.3** Dataset after concept drift

Now, example of concept drift would be if FA students switched drawing boards for tablets and FIT students started to attend drawing courses en masse, hence carrying drawing boards. Our dataset could then look like in the table 1.3.

In this case, the probabilities are

$$\begin{aligned}
 P(B) &= \frac{1}{2} = 0.5 \\
 P(FA) &= \frac{3}{5} = 0.6 \\
 P(B|FA) &= \frac{1}{6} = 0.1\bar{6} \\
 P(FA|B) &= \frac{1}{5} = 0.2
 \end{aligned} \tag{1.2}$$

We can see that probability of student being from FA based on carrying drawing boards (equation 1.2) has decreased radically. Note that distribution of predictor variable, i.e. $P(B)$ did not change, but the relationship between predictor variable and the target has changed.

1.1.2 Multivariate data drift

Until now, when we were talking about the data drift, we talked about **univariate data drift**. What that means is that the drift is happening only in one dimension, i.e. only one predictor variable undergoes the drift. This kind of drift is easier to detect by more common statistical methods, which will be described later.

More troublesome pitfall we can face in the data quality is **multivariate data drift**. The reason why this is worse than univariate drift, as we describe later on, is that we have to look not only for changes in individual variables but also for changes in their relationships. No matter how much we would like to have all of our predictor variables truly independent between themselves, in real world this is usually not the case. In the event of multivariate data drift, probability distributions of individual variables may not change, but the *relationship* between them does. Consider the previous example with classifying students from table 1.1. Now, we could use a well known logistic regression for this problem. Using *has_drawing_boards* and also until now not used column *gender*, we get mean accuracy of 0.9 on the training data. Pearson correlation between gender and whether person is carrying the drawing boards is in this case circa 0.19. Now, imagine that with some new production data all FA students would switch to tablets, but at the same time all women started to attend drawing courses and carrying boards. In this case, our production dataset would look like on the table 1.4.

faculty	gender	has boards
FIT	F	1
FIT	M	0
FIT	M	0
FIT	M	0
FA	F	1
FA	F	1
FA	F	1
FA	F	1
FA	M	0
FA	M	0

■ **Table 1.4** Dataset after multivariate drift

If we compute the probabilities as in previous example, we can see that none of them changed.

$$P(B) = \frac{1}{2} = 0.5$$

$$P(FA) = \frac{3}{5} = 0.6$$

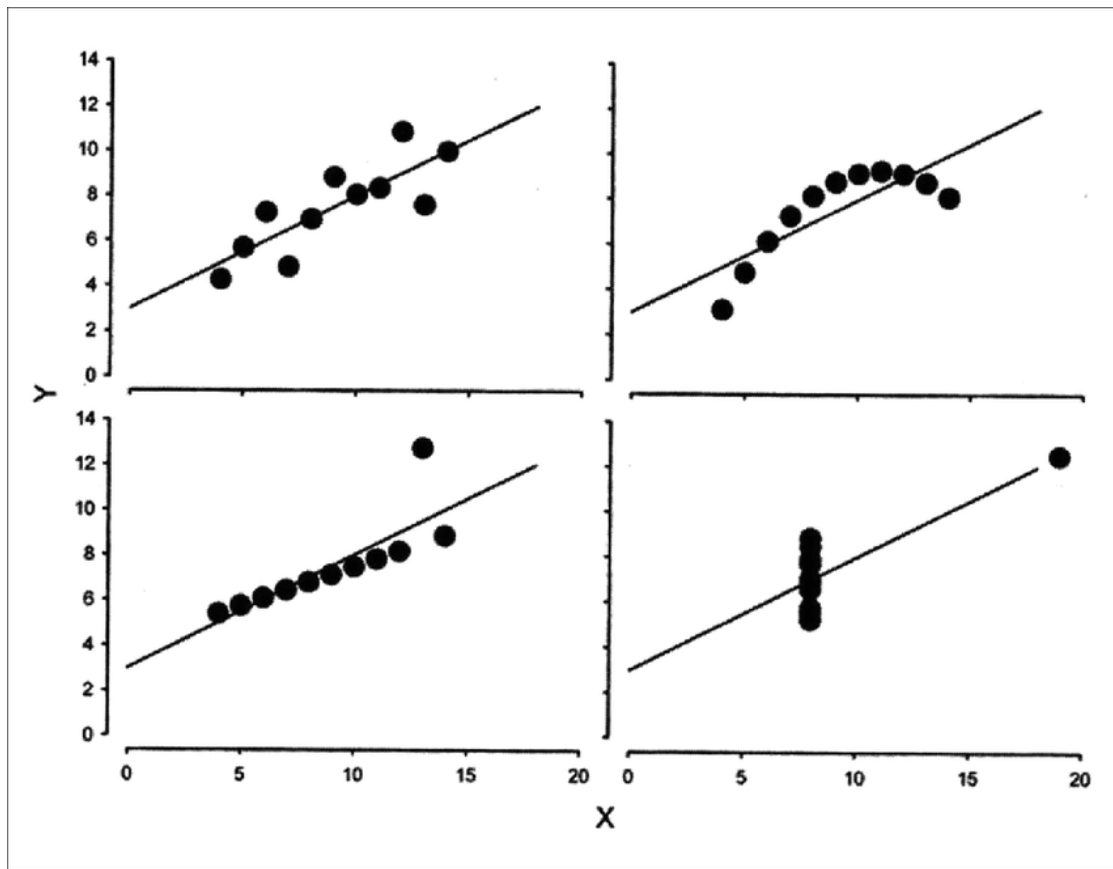
$$P(B|FA) = \frac{2}{3} = 0.\bar{6}$$

$$P(FA|B) = \frac{4}{5} = 0.8 \tag{1.3}$$

Relationship between carrying drawing boards and attending Faculty of Architecture (equation 1.3) remained the same due to distribution of genders between faculties. But, if we look at the Pearson correlation between gender and whether person is carrying drawing boards, we can see that it is now equal to 1. If we now used the linear regression trained on original dataset we get mean accuracy of 0.7, compared to 0.9 on original data.

This example may lead to a false impression, that correlation between variables is the only culprit here and checking it with every new dataset should be enough to solve the issue presented. A counterexample to that is the Anscombe's quartet [3]. It consists of four datasets, each having eleven (X, Y) pairs. If we used standard statistics to describe the datasets, we would see that each of them has identical or almost identical mean of x 's, mean of y 's, equation of linear regression line, sum of squares and correlation between x and y . But, if we look at them graphically, we can see that there are considerable differences between them, as shown on figure 1.1.

As we can see, multivariate drift can lead to decrease in model quality. And in some cases, like the one in the example, it can be difficult to notice the culprit, due to distributions of individual variables remaining the same.



■ Figure 1.1 Anscombe's quartet [4]

When researching about data quality tools, three most used Python libraries stood out. Although Great Expectations was chosen by us as the library that will be extended, we did a comparison with its alternatives. This chapter provides a short summary.

2.1 Great Expectations

Great Expectations is an open-source Python library which helps with validation of production datasets against chosen criteria. The way it does that is by creating an *Expectation*.

The Expectation is an assertion in the form of a simple Python method. GX then uses this assertion to validate new data. The Expectation receives data as input and outputs success or failure. Some Expectations are bundled in the library and many more custom ones were added by users. They range from simple ones like *expect_column_mean_to_be_between* to more complicated ones like *expect_day_sum_to_be_close_to_equivalent_week_day_mean*.

The library is in the development at the time of writing this thesis. During the assignment of the topic, it was in its earlier version. Although already packed with many useful features and more on the way, some features like multivariate drift detection were not present and they were not on the roadmap. Hence the motivation behind this thesis. However, during the course of writing, contributors presented many imported changes into the library and its development sped up considerably. As of December 2023, they are planning to release version 1.0. Because of that, temporary hold has been put on pull requests, due to the breaking changes taking place. Therefore, it would not be feasible to keep track of changes in the library and constantly updating our code. So we decided to fix the version of Great Expectations on 0.16.16. This is not too much behind the latest version, as of December 2023 and it has a second latest version of documentation.

2.2 NannyML

NannyML [5] is an open-source Python library which helps to estimate model performance in production and to detect the data drift in selected datasets. Compared to Great expectations, this tool is more focused on monitoring model performance, then data quality. However, its data drift detection supports also multivariate drift, which GX does not, at the time of writing of this thesis. One of its interesting features is also ability to link between changes in model performance and data drift alerts, which allows to dismiss alerts which do not correspond to lowered data quality.

2.3 Evidently

Evidently [6] is another open-source Python library, used for testing and monitoring of machine learning models in production. It consists of three main modules:

1. **Tests** - Tests module allows creation of a test suite on the given batch of data. Evidently provides wide range of tests, which can check for data stability, presence of data drift and data quality. It can also compare user's model performance to dummy model, to verify its quality.
2. **Reports** - Reports module can calculate and visualise rich variety of ML metrics. It can provide HTML, JSON, or Python dictionary output.
3. **ML monitoring dashboards** - ML monitoring dashboard can be self-hosted and used to visualise several metrics over time. As advertised by Evidently, more than a hundred different metrics can be tracked at the same time.

Compared to GX and NannyML, this library focuses more on model monitoring and visualisation of metrics over time. It has a wide range of features, however at the time of writing this thesis it does not support detection of the multivariate drift.

Methods discussion

3.1 Univariate drift detection

Before we lunge at the multivariate drift detection, machine learning models in hand, let us take a closer look at the statistical methods used for univariate drift detection. It will help us understand the possible approaches used for this kind of problems and realise, whether we can leverage them with multivariate drift.

3.1.1 Kolmogorov–Smirnov test

Two-sample Kolmogorov–Smirnov test (KS test) compares two samples of data and tests the null hypothesis, that they originate from the same (unspecified) distribution or generating process. It is a very common method for analysing data in search for the data drift.

The test statistics

$$D_s = \max |F_n(x_1) - F_m(x_2)|$$

looks for the largest absolute value difference between the empirical cumulative distribution functions of samples of n observations of x_1 and m observations of x_2 . We reject the null hypothesis at level α if $D_s > \sqrt{-\ln \frac{\alpha}{2} \cdot \frac{n+m}{2n \cdot m}}$ [7].

3.1.2 Kullback-Leibler divergence

Kullback-Leibler divergence (KL divergence) is a non-symmetric measure of difference between two probability distributions. It is defined as

$$D_{KL}(p(x)||q(x)) = \sum_{x \in X} p(x) \ln \frac{p(x)}{q(x)}$$

where $p(x)$ and $q(x)$ are probability distributions of discrete random variable x . Its continuous form is defined as

$$D_{KL}(p(x)||q(x)) = \int_{-\infty}^{\infty} p(x) \ln \frac{p(x)}{q(x)} dx$$

As mentioned before, KL divergence is non-symmetric, meaning that $D_{KL}(p(x)||q(x)) \neq D_{KL}(q(x)||p(x))$. This has its advantages and disadvantages, but nevertheless it has to be taken

into account, when using KL divergence to detect data drift. This also means that it cannot be used as a *metric* of a metric space.

3.1.3 Population Stability Index

Problems with non-symmetry of KL divergence can be solved by using Population Stability Index (PSI), which is a symmetric derivation of KL divergence. We can define it as

$$PSI(p(x), q(x)) = D_{KL}(p(x)||q(x)) + D_{KL}(q(x)||p(x))$$

which can be rewritten as

$$PSI(p(x), q(x)) = \sum_{x \in X} (p(x) - q(x)) \ln \frac{p(x)}{q(x)}$$

or in its continuous form as

$$PSI(p(x), q(x)) = \int_{-\infty}^{\infty} (p(x) - q(x)) \ln \frac{p(x)}{q(x)} dx$$

This measure is widely used, among others by banks to detect shift between training and production data [8].

3.2 Multivariate drift detection

3.2.1 Binary classifier

One way to go about the data drift is to look at it as a “significant enough” change in the data. It may be difficult for the person analysing it to detect this change by simply looking at the relevant statistics. But a machine learning model might do better. We could train a binary classifier that will try to classify, whether given sample comes from the training data, or current production data. If this classifier has a low accuracy, resembling a random choice, it is probable that no drift has occurred and that these data samples come from the same generating process. But, if accuracy of the model is high, it means that it can correctly identify whether the sample comes from the training dataset, or the production one. This in turn indicates, that there is some distinguishable feature that sets them apart, a “change”, or as we call it - a *drift*. Threshold accuracy, after which can be said that the potential drift has occurred can be determined by bootstrapping. In our scenarios, we used 60% as the threshold value. This method could also be used for univariate drift detection, however it is not so common due to existence of abovementioned simpler methods.

3.2.2 Clustering

Using K-means, DBSCAN or other clustering method we can find clusters in the reference dataset and current dataset [9]. To compare these clusters, we use Silhouette score [10]. Silhouette score is a metric which estimates goodness of a clustering method. It ranges from -1 to 1 . 1 mean clusters are well differentiated, -1 means that clusters are incorrect.

We can leverage this for data drift detection. By computing silhouette score for both reference and tested dataset and subtracting them, we obtain a metric estimating how good the clusters obtained from reference represent the tested dataset.

As a threshold, value of -0.1 was obtained by bootstrapping.

3.2.3 Gaussian Mixture Models(GMM)

Gaussian Mixture Models (GMM) is a probabilistic model that assumes all the data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters [11] [12]. The presence of data drift can be detected by comparing parameters of Gaussian distributions of training data to production data [9].

This method has its limitations. Assumption that data come from Gaussian distribution is quite strong and can lead to inferior results if erroneous [13].

3.2.4 Principal Component Analysis (PCA)

In order to detect the multivariate drift in the dataset, we can try to leverage its principal components. They capture linear correlations between features, therefore may help us with perceiving changes between them. An approach we can use is the following one:

- Train and fit PCA on referential dataset
- Use reverse PCA to reconstruct the referential dataset
- Compute a **reconstruction error**
- For each new dataset in production, use PCA trained on referential dataset and fitted on tested one to compute the reconstruction error and compare it with the reference

Reconstruction error is in this case some sort of distance between original dataset and the one obtained by applying PCA and reverse PCA transformations in order. Commonly used measure for this distance is **Euclidean distance**.

PCA is in principle sensitive to outliers. Their addition to data can significantly change the principal components computed by this method. This in turn would cause a lot of false positives during drift detection. As a way to solve this problem we propose to perform data stabilisation to some extent before applying PCA. Values further than three standard deviations from mean can be considered as outliers and be removed from both referential and tested datasets.

For the purpose of the testing, we used bootstrap to determine the threshold as 1.3.

3.3 Methods comparison

In order to compare these methods and find their strengths and weaknesses, we prepared a set of testing scenarios. Using a script we generated example datasets on which we then simulated the drift. The script defines a generating process for the data. Drift is therefore simulated by specific changes in this process. This approach was chosen over fixed datasets, because it should be better at mimicking a real life use case. In such event, new data is often obtained in batches over a given period, with some generating process behind the data values.

During the creation of these scenarios we had to make certain decisions. Firstly, we decided to use artificially generated data instead of real world one. The reason for this is that we can tailor the data in such a way, that we can observe how compared methods fare in different specific cases. Next, we chose normal distribution as the distribution of the data. It is one of the primary distributions and its properties are suitable for showcasing the scenarios. Choosing one specific distribution for the data also reduces the number of factors that can affect the results. Lastly, we chose smaller datasets exhibiting certain traits or edge cases that we want to observe, instead of large datasets consisting of many features, that would be harder to analyse.

Main tool used for implementing tested methods and necessary automation was scikit-learn [11].

The datasets consists of three variables, x_1, x_2, x_3 . In the starting configuration, x_1 and x_2 are independent random variables drawn from normal distribution, whereas x_3 is a linear combinations of them. Algebraically written as:

$$x_1 \sim N(10, 3)$$

$$x_2 \sim N(20, 5)$$

$$x_3 = 12 \cdot x_1 - 6 \cdot x_2 + \epsilon$$

where $\epsilon \sim N(0, 1)$ is a random variable representing noise. Main focus was on variables x_2 and x_3 , the last one is used to control the generating process.

We used seven scenarios to compare the methods:

1. No drift: used to check for false errors
2. Univariate drift: change in the mean
3. Univariate drift: change in the standard deviation
4. Univariate drift: change in both mean and standard deviation
5. Multivariate drift: change in coefficients of linear combination of variables - changed sign
6. Multivariate drift: change in coefficients of linear combination of variables - changed magnitude
7. No drift, only outliers were added to dataset

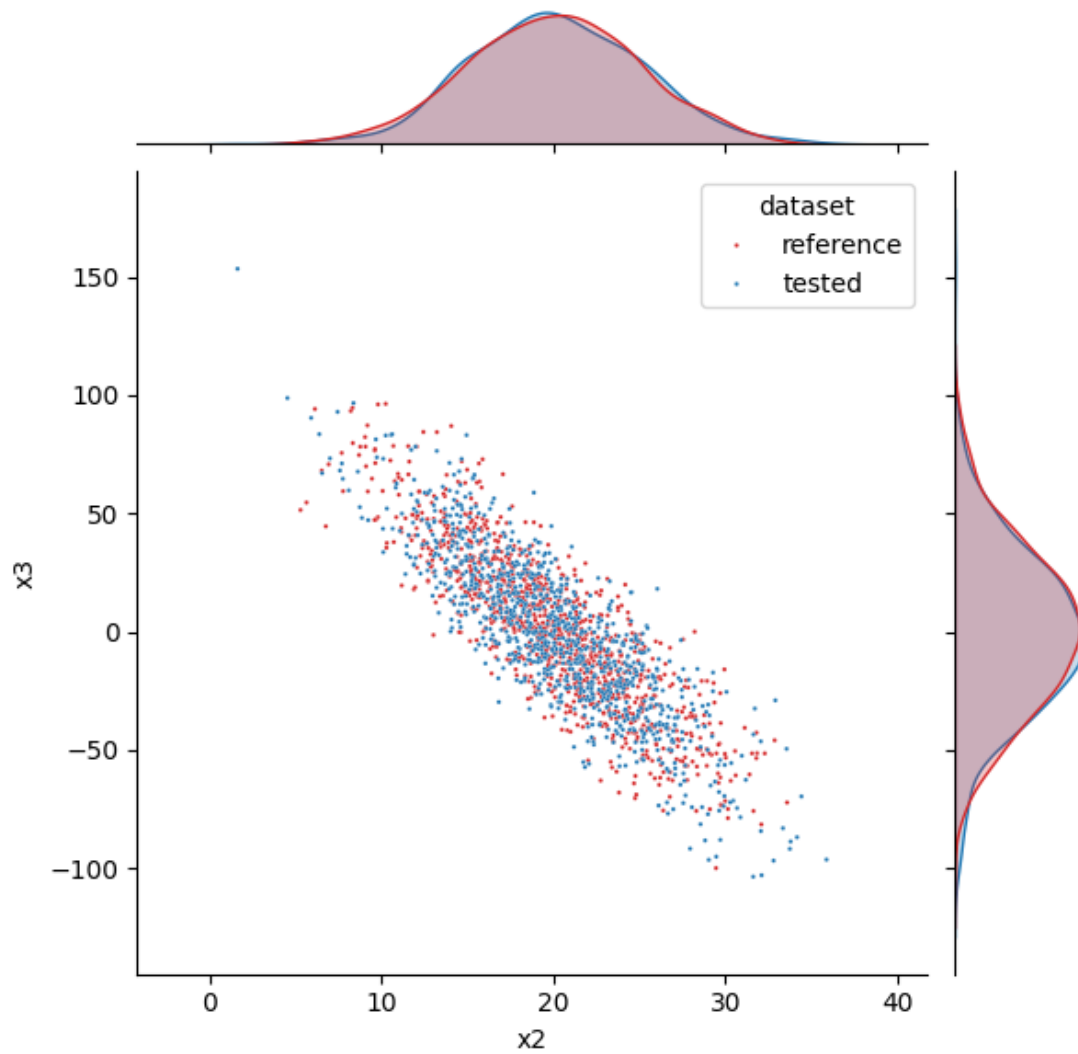
Following small sections are devoted to the individual scenarios. Each contains a brief description, table with the results and graphical visualisation of the data in that particular case.

Table with the results contains three columns.

1. Method - the name of the method testes
2. Value - value computed by the method, that decides whether the drift has occurred. In the row with Kolmogorov–Smirnov test, value corresponds to the lowest p-value calculated from all features. Binary classifier values represents accuracy of the model. In the PCA row, value is the ratio between the reconstruction errors on referential and tested dataset. With clustering, value shows difference between silhouette score computed on tested dataset and the silhouette score computed on referential data.
3. Data drift detected - final verdict obtained by comparing value of the method with a given threshold.

3.3.1 No drift

This scenario tests how prone are individual methods to false positives. Tested dataset was in this case generated from the same generating process as the referential one.



■ **Figure 3.1** Tested and referential dataset comparison without drift

Method	Value	Data drift detected
K-S test	0.148	False
Binary classifier	0.507	False
PCA	0.981	False
PCA w. stabilisation	0.966	False
Clustering	-0.033	False

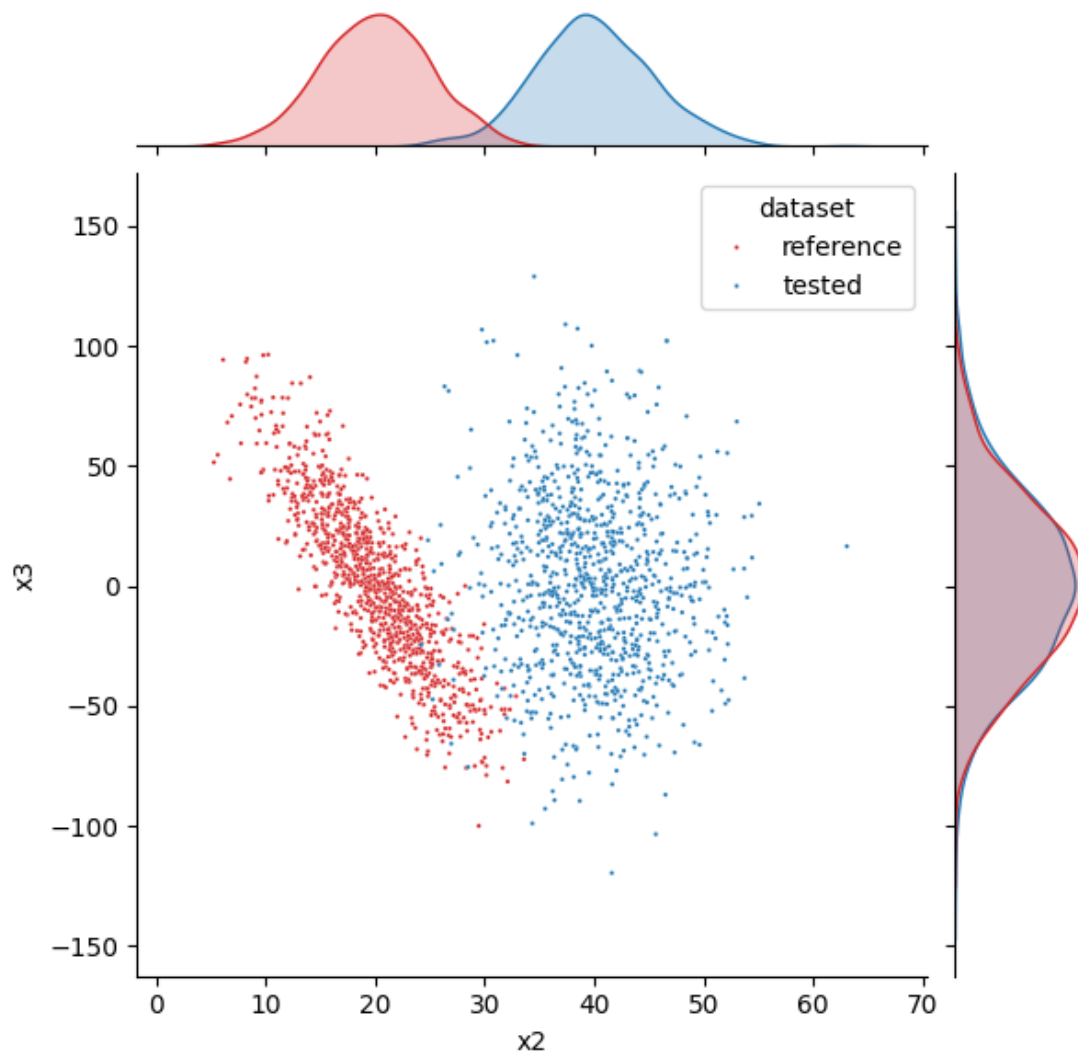
■ **Table 3.1** No drift

3.3.2 Univariate shift - mean

In this scenario, variable x_2 has undergone the univariate shift, specifically the change of its mean.

$$x_1 \sim N(40, 1)$$

The scenario tests the ability of methods to detect a change in the magnitude of the features.



■ **Figure 3.2** Tested and referential dataset comparison with shift in mean

Method	Value	Data drift detected
K-S test	0.000	True
Binary classifier	0.976	True
PCA	3.088	True
PCA w. stabilisation	3.076	True
Clustering	-0.138	True

■ **Table 3.2** Univariate drift - shift in mean

3.3.3 Univariate shift - standard deviation

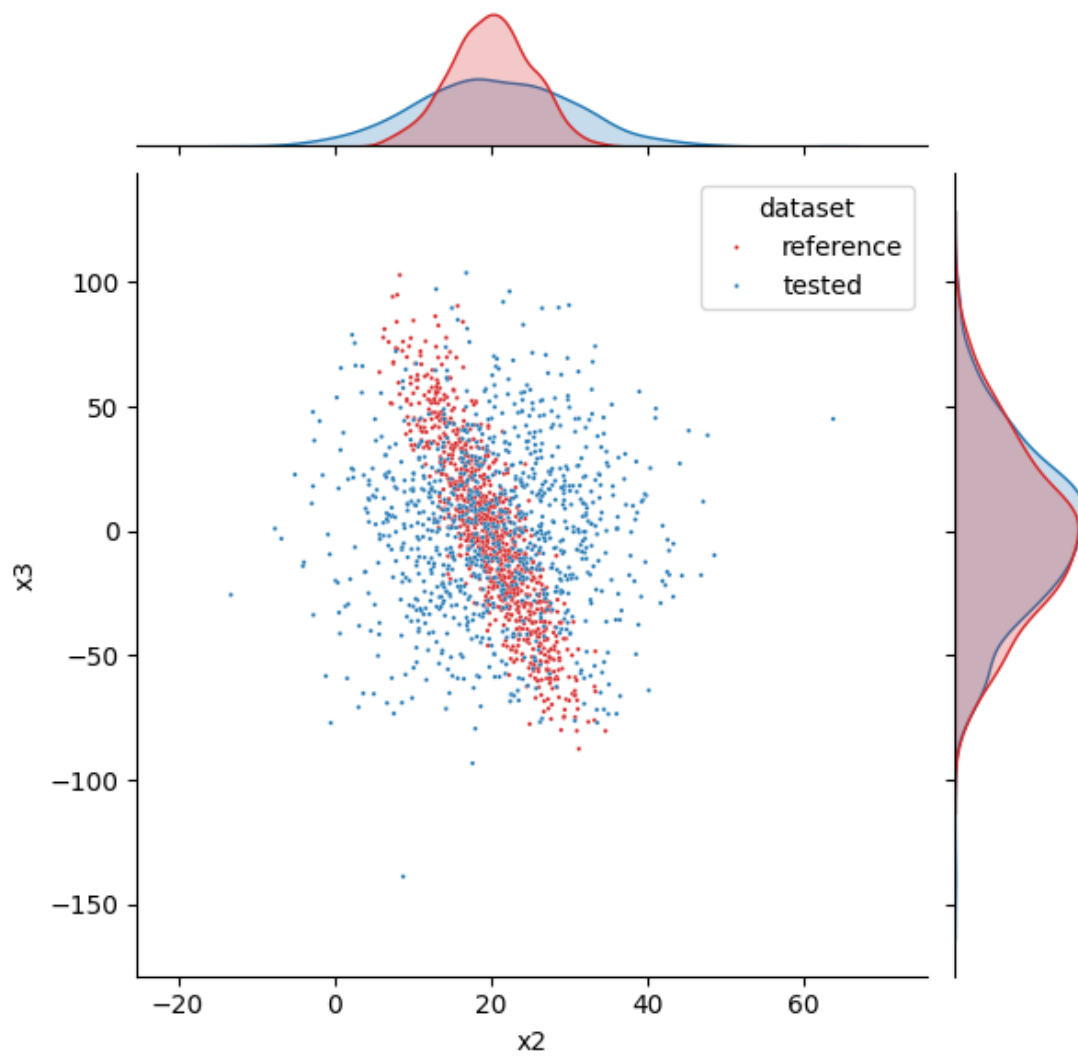
In this case, the standard deviation of x_2 has changed.

$$x_1 \sim N(20, 10)$$

This scenario tests the sensibility of the methods to change in spread of the features.

Method	Value	Data drift detected
K-S test	0.000	True
Binary classifier	0.547	False
PCA	1.566	True
PCA w. stabilisation	1.549	True
Clustering	-0.063	False

■ **Table 3.3** Univariate drift - shift in standard deviation

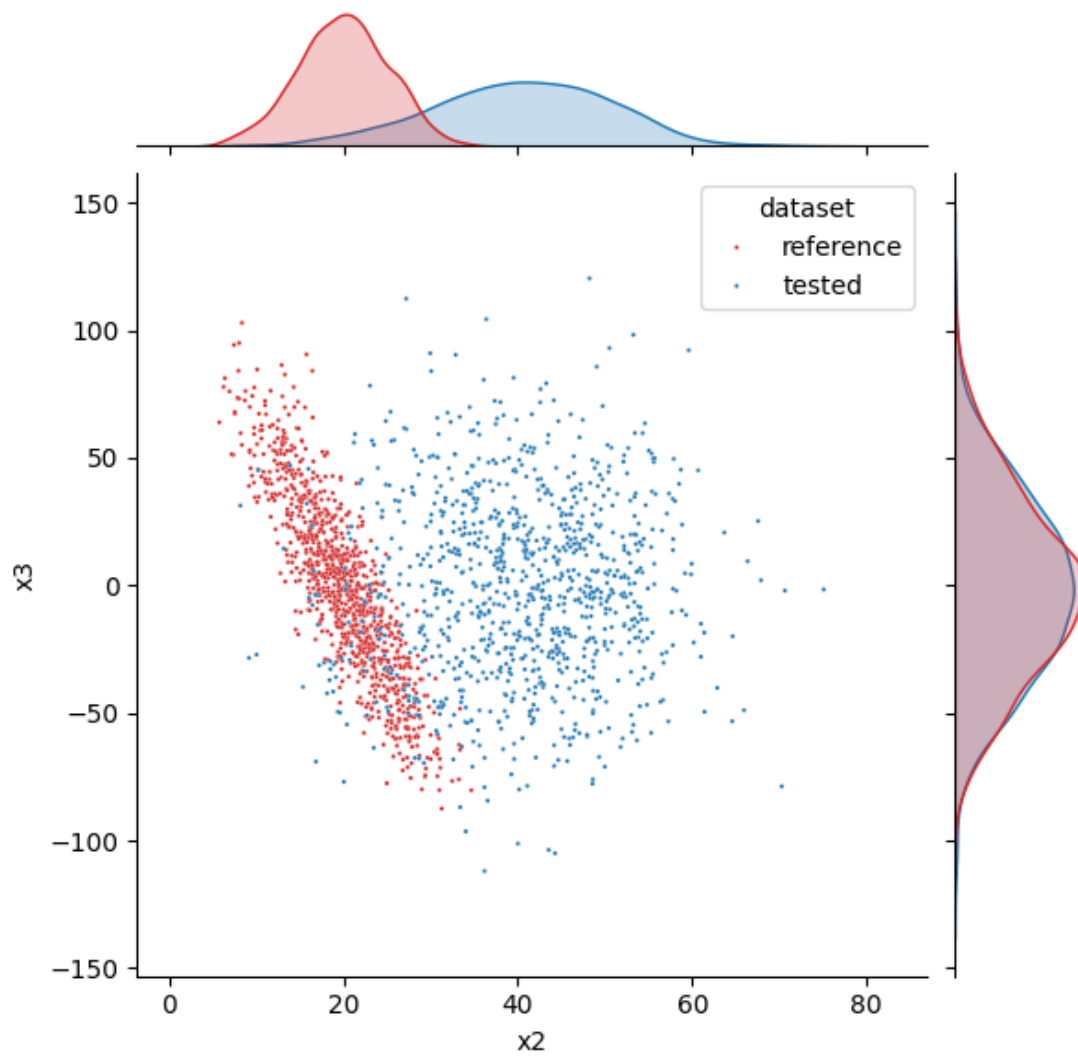


■ **Figure 3.3** Tested and referential dataset comparison with shift in standard deviation

3.3.4 Univariate shift - mean and standard deviation

This scenario is a combination of previous two, where x_2 undergoes a change in both mean and standard deviation.

$$x_1 \sim N(40, 10)$$



■ **Figure 3.4** Tested and referential dataset comparison with shift in mean and standard deviation

Method	Value	Data drift detected
K-S test	0.000	True
Binary classifier	0.919	True
PCA	3.386	True
PCA w. stabilisation	3.363	True
Clustering	-0.131	True

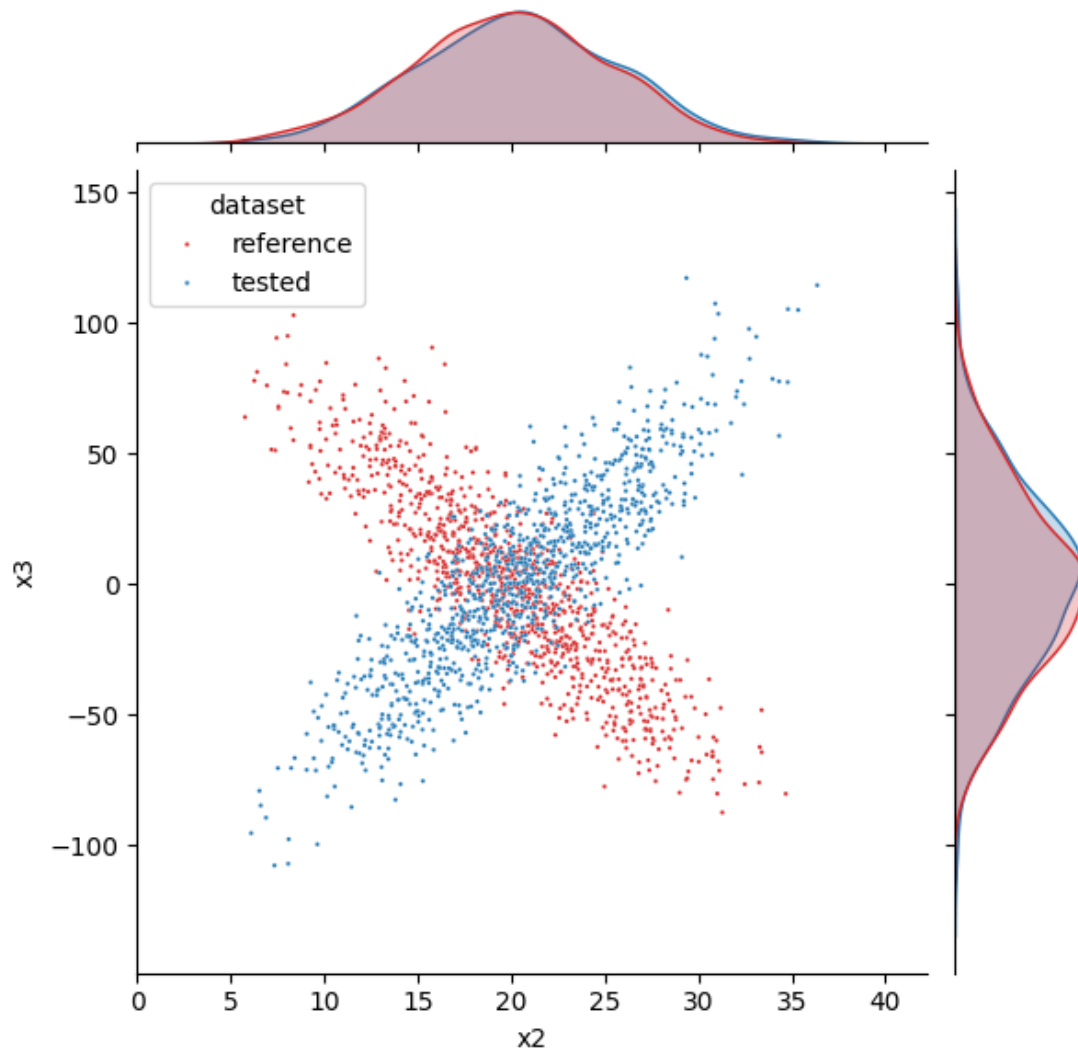
■ **Table 3.4** Univariate drift - shift in mean and standard deviation

3.3.5 Multivariate shift - switching signs

In this case, relation between variables has changed. To make the change less noticeable in terms of changes in distribution, only signs in its equation were flipped.

$$x_3 = -12 \cdot x_1 + 6 \cdot x_2 + \epsilon$$

Now it correlates negatively with x_1 and positively with x_2 . However, shape of its distribution remained roughly the same, as seen on figure 3.5



■ **Figure 3.5** Tested and referential dataset comparison with multivariate shift - just the sign

Method	Value	Data drift detected
K-S test	0.078	False
Binary classifier	0.501	False
PCA	2.173	True
PCA w. stabilisation	2.117	True
Clustering	-0.029	False

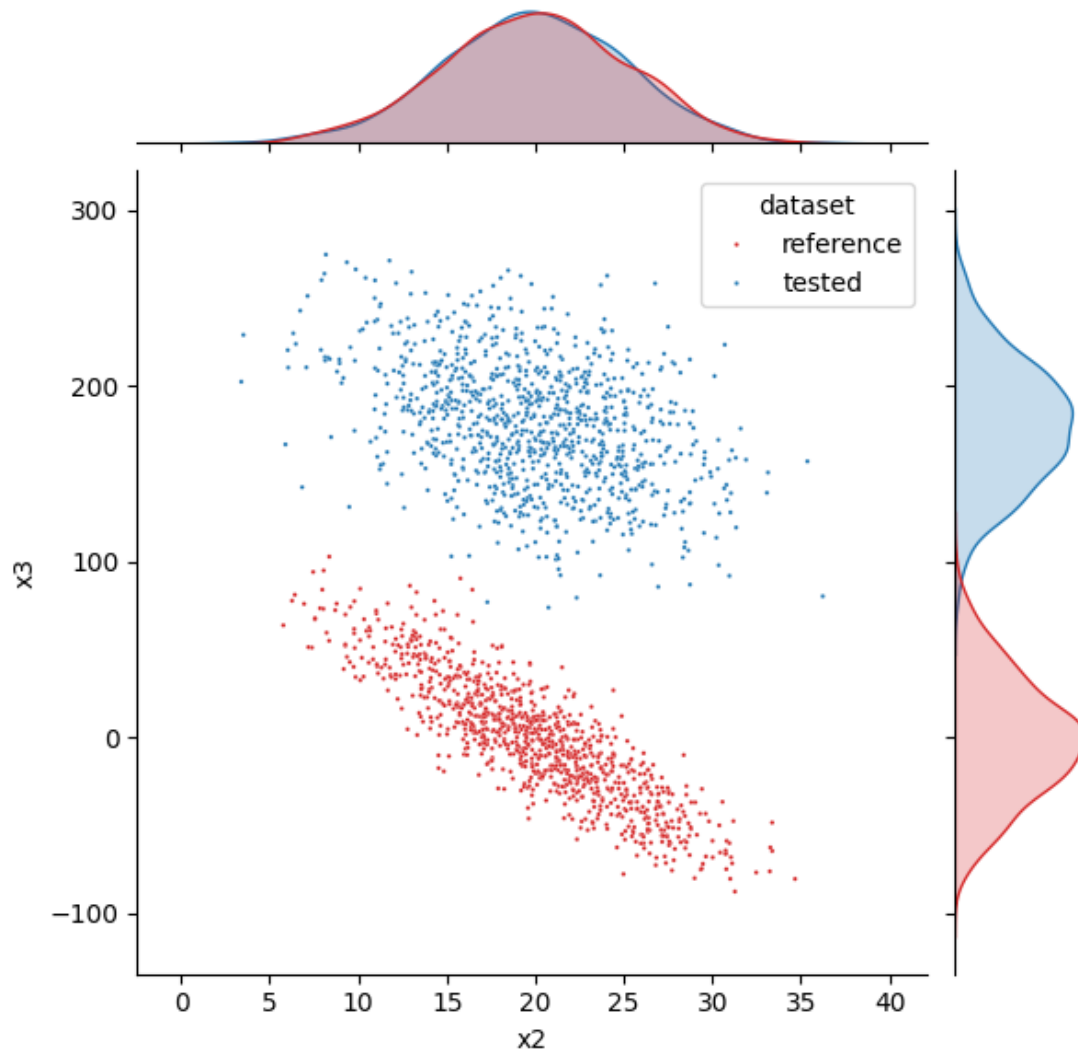
■ **Table 3.5** Multivariate shift - just the sign

3.3.6 Multivariate shift - change in scale

In this scenario, correlation between x_3 and x_1 has increased, while correlation between x_3 and x_2 has decreased.

$$x_3 = 24 \cdot x_1 - 3 \cdot x_2 + \epsilon$$

While this change occurs in relationship between variables, it is also reflected in the distribution of x_3 as seen on figure 3.6.

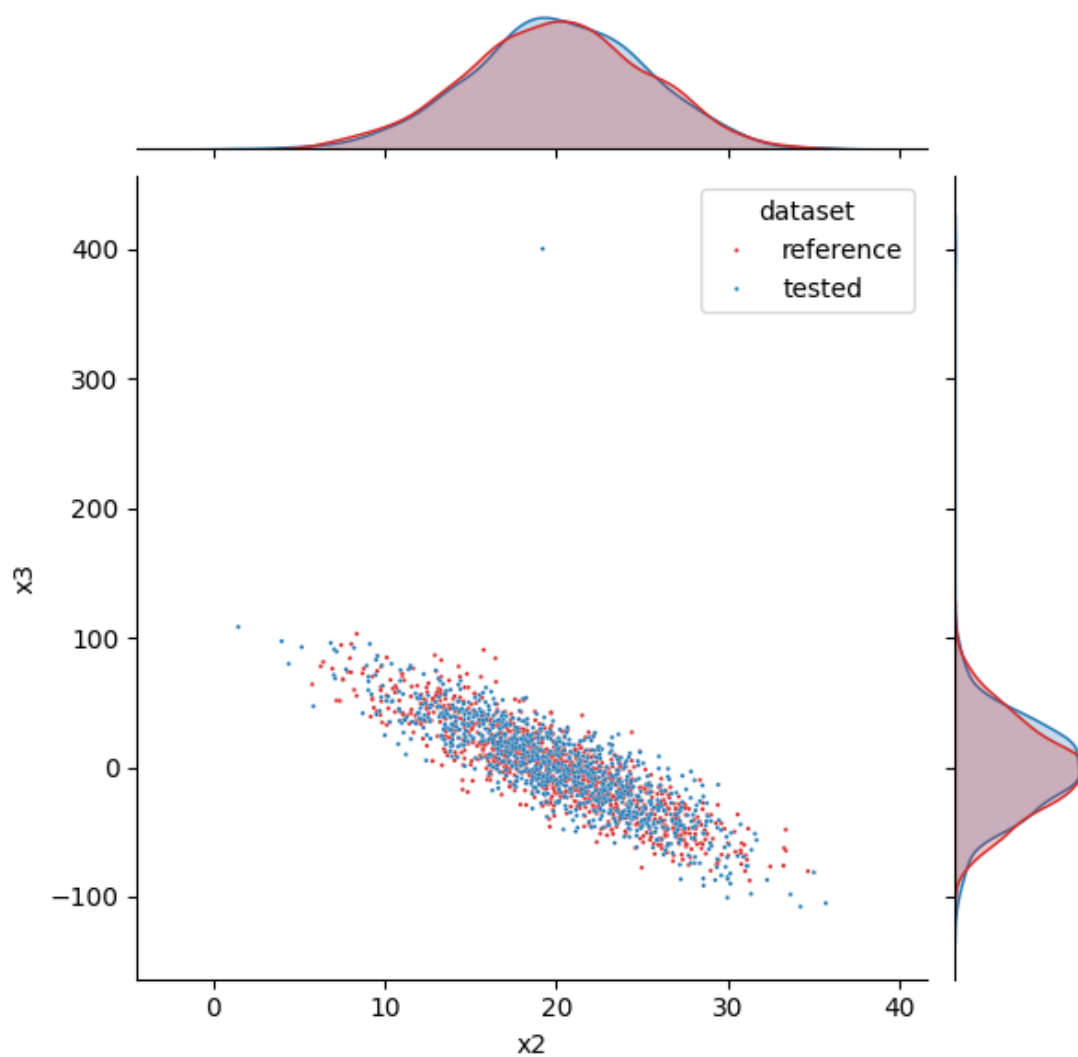


■ **Figure 3.6** Tested and referential dataset comparison with multivariate shift - just the scale

Method	Value	Data drift detected
K-S test	0.000	True
Binary classifier	1.000	True
PCA	3.852	True
PCA w. stabilisation	3.839	True
Clustering	-0.122	True

3.3.7 Adding outliers

In this scenario no drift occurred, however outliers were added to the dataset. Detection of outliers also belongs to the quality of data but in most cases it should not trigger data drift alert with the methods used.



■ **Figure 3.7** Tested and referential dataset comparison with adding outliers

Method	Value	Data drift detected
K-S test	0.181	False
Binary classifier	0.482	False
PCA	1.027	False
PCA w. stabilisation	0.962	False
Clustering	-0.012	False

■ **Table 3.6** Adding outliers



Chapter 4

Results

From the results we can make some observations.

Kolmogorov–Smirnov test works good for univariate drift. Since it is possible to use it only in one dimension, when it comes to multivariate drift, it can only detect the drift if a change in relationship between variables causes significant change to the distribution of one variable.

In our scenarios, logistic regression was used for binary classifier. We used a naive implementation in which model used all predictors to determine whether data point belongs to reference or tested batch. It can be seen that this approach has its limitations. While model was sufficiently good at detecting changes in the mean, it failed to detect changes in standard deviation. More sophisticated approaches could be used, for example using only certain predictors or using altogether different classifier. On the other hand this might require better understanding of the data context which carries its own disadvantages.

PCA worked well in almost all cases. It was reliable for detection of both univariate and multivariate drifts. Downside of this method showed itself when outliers were added to the dataset. As PCA is sensitive to the outliers in principle, this caused a lot of false positives of drift. One of the solutions to this problem could be the removal of the outliers (values further than three standard deviations from mean). When tested, PCA showed better results after this stabilisation.

Clustering performed similarly to binary classification. And as mentioned with the classifier also here are possible approaches which could lead to better performance. KMeans algorithm was used in our test, with a fixed default number of clusters. However, more complex analysis of the data could be done using silhouette analysis to choose a better value tailored to the dataset.

Implementation

5.1 Expectation

Based on performed comparison of methods, Principal component analysis showed the best results in detecting univariate and multivariate data drift. Binary classifier and clustering could be improved and therefore yield better performance, however they possess a disadvantage when it comes to computational time. They would also need more data preprocessing. Mainly due to these reasons PCA was chosen as the method for new Expectation.

We used *BatchExpectation* template provided by GX as a base for the new Expectation. There are two main parts of the implementation, **metric** and **validation**. Metric is a function called to compute desired values out of given batch of data. In our case this means to compute the reconstruction error. In validation part, this error is then compared to reference value and the result is the output of the expectation.

5.2 Profiler

One of the goals of this thesis was a creation of a data profiler, which in this context means automatic tool that based on some rules analyses given data and chooses a set of expectations, which will form a testing suite. At the time of creation of the assignment, only a simple version of this tool was present in GX. However, during the writing of the thesis, Great expectations library was significantly extended and improved. One of the changes was also an addition of a rule-based profiler. This gives user a highly configurable tool which allows choosing custom rules, thresholds and expectations for the given data. [14]. It consists of three components:

1. Domain builder - Inspects the provided data and creates a set of domains (in this context types of features) for which expectations will be assembled
2. Parameter builder - Based on the data it compiles a dictionary of data metrics, which will be passed to the expectations as parameters
3. Expectation Configuration Builders - Combines results compiled by previous two builders and assembles configurations for suitable expectations

Since this tool allows basically the same functionality as the profiler proposed in the thesis assignment, pursuit of this particular goal was abandoned. Instead, focus was transferred to the comparison of individual methods for drift detection.



Chapter 6

Conclusion

This thesis concerned itself with data drift detection. Main focus was on multivariate drift between continuous variables. During the work on this thesis, we explored an open-source Python tool for data quality testing - Great Expectations.

Then we compared various drift detection methods using multiple scenarios which tested ability of the methods to detect changes in mean, standard deviation and relationship between predictors. We also tested the sensitivity of the methods to outliers and whether they are prone to false alarms. Based on the results we chose the most suitable method - Principal component analysis. In the end we implemented this method to the structured Expectation in the GX library.

The last goal of this thesis was the profiler, however due to it already being implemented in the GX library, this goal was abandoned.

Bibliography

1. GONG, Abe; CAMPBELL, James; GREAT EXPECTATIONS. *Great Expectations*. [N.d.]. Available also from: https://github.com/great-expectations/great_expectations.
2. TSYMBAL, Alexey. The Problem of Concept Drift: Definitions and Related Work. 2004. Available also from: https://www.researchgate.net/publication/228723141_The_Problem_of_Concept_Drift_Definitions_and_Related_Work.
3. ANSCOMBE, F. J. Graphs in Statistical Analysis. *The American Statistician* [online]. 1973, vol. 27, no. 1, pp. 17–21 [visited on 2024-01-10]. ISSN 00031305. Available from: <http://www.jstor.org/stable/2682899>.
4. BRANCH, Marc. malignant side effects of null-hypothesis significance testing. *Theory & Psychology*. 2014, vol. 24, pp. 256–277.
5. *NannyML (release 0.10.2)* [<https://github.com/NannyML/nannyml>]. 2023. NannyML, Belgium, OHL.
6. SAMUYLOVA, Elena; EVIDENTLY AI. *Evidently*. [N.d.]. Available also from: <https://github.com/evidentlyai/evidently>.
7. WILKS, Daniel S. (ed.). Chapter 5 Hypothesis testing. In: *Statistical Methods in the Atmospheric Sciences*. Academic Press, 1995, vol. 59, pp. 114–158. International Geophysics. ISSN 0074-6142. Available from DOI: [https://doi.org/10.1016/S0074-6142\(06\)80041-0](https://doi.org/10.1016/S0074-6142(06)80041-0).
8. YURDAKUL, Bilal; NARANJO, Joshua. Statistical Properties of the Population Stability Index. 2019, vol. 14, no. 4. Available also from: <https://ssrn.com/abstract=3783305>.
9. TRUEFOUNDRY. *A Guide to drift tracking* [online]. TrueFoundry, Feb 2023 [visited on 2024-01-10]. Available from: <https://blog.truefoundry.com/guide-to-drift-tracking/>.
10. SHUTAYWI, Meshal; KACHOUIE, Nezamoddin N. Silhouette Analysis for Performance Evaluation in Machine Learning with Applications to Clustering. *Entropy (Basel)*. 2021, vol. 23, no. 6.
11. PEDREGOSA, F.; VAROQUAUX, G.; GRAMFORT, A.; MICHEL, V.; THIRION, B.; GRISEL, O.; BLONDEL, M.; PRETTENHOFER, P.; WEISS, R.; DUBOURG, V.; VANDERPLAS, J.; PASSOS, A.; COURNAPEAU, D.; BRUCHER, M.; PERROT, M.; DUCHESNAY, E. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*. 2011, vol. 12, pp. 2825–2830.
12. SCIKIT-LEARN. *Gaussian mixture models* [online]. scikit-learn, 2011 [visited on 2024-01-10]. Available from: <https://scikit-learn.org/stable/modules/mixture.html#gaussian-mixture-models>.

13. KASA, Siva Rajesh; RAJAN, Vaibhav. Avoiding inferior clusterings with misspecified Gaussian mixture models. *Scientific Reports*. 2023, vol. 13, no. 1, p. 19164. ISSN 2045-2322. Available from DOI: [10.1038/s41598-023-44608-3](https://doi.org/10.1038/s41598-023-44608-3).
14. GONG, Abe; CAMPBELL, James; GREAT EXPECTATIONS. *Great Expectations*. [N.d.]. Available also from: https://github.com/great-expectations/great_expectations/tree/develop/great_expectations/rule_based_profiler.

Contents of the attachment

Readme.md.....	short description of the attachment
└ GX_library	
└ great_expectations	source code of the implementation
└ data	
└ methods_comparison	python scripts and jupyter notebooks containing research
└ thesis.pdf	text of the thesis in PDF
└ thesis.zip.....	source code of the thesis in L ^A T _E X
└ requirements.txt.....	requirements for pip